Computer Science and Application 计算机科学与应用, 2025, 15(3), 8-19
Published Online March 2025 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2025.153053

Hans汉斯

数字纹理图像生成算法研究

赵珊珊，曹　鹏*

北京印刷学院信息工程学院，北京

收稿日期：2025年2月2日；录用日期：2025年2月28日；发布日期：2025年3月12日

摘　要

本研究提出一种利用参数方程生成数字纹理图像的方法。该算法通过灵活运用参数方程，生成各类矢量图元(如半圆矢量图元)以构建基础纹理图像，同时对图元进行随机平移或旋转操作，增强纹理的随机性。在对基础图像进行字符化时，从多种类型的总字符集随机选取字符，最终生成兼具特殊艺术效果与卓越防伪特性的数字纹理图像。实验结果有力证明，此算法在生成的数字纹理图像不仅艺术效果独特，且凭借高度的唯一性和随机性，极大提升了防伪性能，为产品提供可靠的防伪保障。

关键词

数字纹理图像，参数方程，随机字符化，艺术效果，防伪特性

Study on Digital Texture Image and Intelligent Reading Algorithm

Shanshan Zhao, Peng Cao*

School of Information Engineering, Beijing Institute of Graphic Communication, Beijing

Received: Feb. 2nd, 2025; accepted: Feb. 28th, 2025; published: Mar. 12th, 2025

Abstract

This research proposes a method for generating digital texture images by utilizing parametric equations. The algorithm, through the flexible application of parametric equations, generates various vector primitives (such as semicircular vector primitives) to construct the basic texture image. Meanwhile, random translation or rotation operations are performed on these primitives to enhance the randomness of the texture. During the process of characterizing the basic image, characters are randomly selected from a diverse set of total character sets. Ultimately, a digital texture image that combines special artistic effects and excellent anti-counterfeiting properties is

*通讯作者。

文章引用: 赵珊珊, 曹鹏. 数字纹理图像生成算法研究[J]. 计算机科学与应用, 2025, 15(3): 8-19.
DOI: 10.12677/csa.2025.153053

generated. The experimental results strongly demonstrate that the digital texture images generated by this algorithm not only possess unique artistic effects, but also, by virtue of their high degree of uniqueness and randomness, significantly enhance the anti-counterfeiting performance, providing reliable anti-counterfeiting protection for products.

Keywords

Digital Texture Images, Parametric Equations, Random Characterization, Artistic Effects, Anti-Counterfeiting Properties

1. 引言

数字纹理图像生成技术在图像处理和计算机图形学等方面具有重要意义，它可广泛存在于通信、安全等领域，具有很高的应用价值[1]。自然界中的物质纹理千差万别，例如人的指纹、树叶的脉络、瓷器的裂纹和纸张的纤维分布等，均具有独一无二且无法重复的特性。正是由于纹理的唯一性和随机性，纹理防伪已成为防伪技术的重要发展方向。

数字纹理图像生成算法的研究历经多年发展，国内外学者在该领域取得了丰硕成果。随着时间的推移，数字纹理图像生成算法逐渐形成了多种不同的类型，其中较为重要的包括基于分形的算法和过程式纹理生成算法。

基于分形的算法是数字纹理图像生成领域的经典方法之一，其核心原理是利用分形几何中局部与整体相似的特性来生成具有丰富细节和自相似结构的纹理[2]。分形几何由 Mandelbrot 于 20 世纪 70 年代正式提出，它打破了传统几何对象的规则性和光滑性假设，揭示了自然界中广泛存在的复杂、不规则但具有自相似性的几何形态[3]。分形理论在多个学科领域都产生了重大影响[4]-[6]。

在数学领域，分形理论的研究主要集中在分形维度、自相似性、迭代函数系统、随机分形、分形插值、分形微分方程等方面。Falconer [6]系统地总结了分形几何学的数学基础，包括 Hausdorff 维数、盒维数、自相似集合、随机分形等。Barnsley [7]深入研究了迭代函数系统(IFS)理论，提出了分形插值、分形图像压缩等重要应用。Hutchinson [8]从测度论的角度研究了自相似集合的存在性和唯一性问题。此外，分形插值函数[9] [10]、分形插值曲面[11] [12]、分形偏微分方程[13] [14]等研究也取得了重要进展。

在计算机科学领域，分形理论被广泛应用于计算机图形学、图像处理、模式识别、人工智能等方面。Barnsley 等[15]系统地总结了分形在计算机图形学中的应用，包括分形地形生成、分形植物建模、分形纹理合成等。Keller [16]研究了分形维数在图像分析和模式识别中的应用，提出了基于盒维数和 Hausdorff 维数的图像分割和特征提取方法。Barnsley 等[17]系统地总结了分形图像压缩的理论和算法，提出了基于 IFS 的编码方法。Pentland [18]将分形维数引入纹理分析和图像分割，开创了基于分形的图像处理方法。此外，分形生成艺术[19]、分形地形生成[20]、分形植物建模[21]等研究也取得了重要进展。

近年来，分形几何学在模拟和分析自然界的复杂形态中展现出了显著的应用价值，特别是在蕨类植物结构的研究方面。Campbell [22]采用分形几何学方法对蕨类植物叶子的形状进行描述和量化分析，展示了蕨类植物叶子数学特性的新视角，并强调了分形几何在解释自然界复杂形态中的应用潜力。Hartvigsen [23]进一步探讨了植物的分形特性，应用分形几何学来分析蕨类植物叶片的形状和结构。这些

研究不仅增进了我们对蕨类植物形态学特性的理解，也展示了分形几何在生物形态分析中的广泛应用。

除了蕨类植物，分形几何学在其他植物形态的研究中也取得了重要进展。Godin 等[24]提出了一种基于分形维数的树木结构复杂性量化方法，揭示了树木分支结构的自相似性和不规则性。这些研究展示了分形几何学在植物形态学研究中的广泛应用，为理解植物形态的生成机制和结构特征提供了新的视角。

基于分形的算法生成的纹理具有独特的优势。其生成的纹理具有无限的细节，无论放大多少倍，都能展现出复杂而精细的结构。分形算法相对简单，主要基于数学迭代公式，不需要大量的样本数据，计算资源消耗相对较少，在一些对实时性要求较高的场景中具有一定的应用价值。但该算法也存在一些局限性。分形算法生成的纹理往往具有较强的规律性和自相似性，在模拟一些随机性较强、缺乏明显自相似特征的自然纹理时，效果可能不够理想。

过程式纹理生成算法是通过数学函数和算法来生成纹理，常见的噪声函数类为 Perlin 噪声。Perlin 噪声是由 Ken Perlin 在 1985 年提出，是一种经典的梯度噪声[25]。它通过在网格上随机生成梯度向量，并根据点与网格顶点的距离进行插值计算，生成平滑、连续且具有一定随机性的噪声值。Perlin 噪声可以生成自然、有机的纹理，如木纹、石纹、云朵等，常用于游戏开发、计算机图形学等领域。其优点是生成的纹理自然、平滑，易于控制和调整；缺点是对于复杂纹理的生成可能需要多层叠加和复杂的参数调整。

尽管数字纹理图像生成算法在过去几十年中取得了显著的进展，但仍然面临着一些挑战。例如，如何生成具有特殊艺术效果和防伪特性的纹理图像以满足不同领域的需求等，是当前研究的重点和难点。

2. 算法设计

2.1. 算法整体流程

本文生成的算法主要包含以下步骤：

a) 生成基础纹理图像：首先根据参数方程生成矢量图元，包括曲线、直线、点、字符、规则图形或不规则图形等[26]，然后对每个矢量图元进行处理，得到具有伪随机特性的基础纹理图像；

b) 字符化基础纹理图像：对 a)中获得的基础纹理图像进行随机字符化，即在设定的总字符集中随机选取字符，将基础图像中的图元均字符化，从而得到具有特殊艺术效果和防伪特性的字符化图像。具体流程图如图 1 所示。

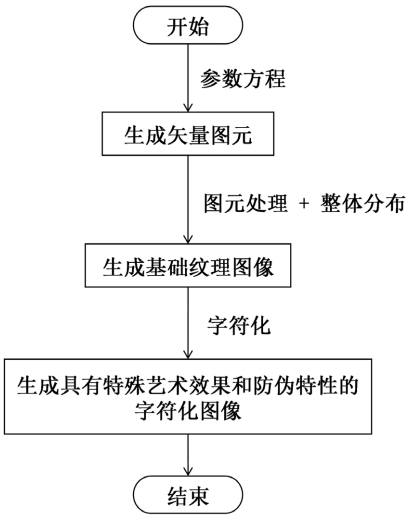


Figure 1. Flowchart of the algorithm
图 1. 算法整体流程图

2.2. 基础纹理图像生成

生活中常见的纹理如纸张纤维、木桌的纹理、瓷砖纹理和布的纹理，千姿百态，各不相同。生成基础纹理图像的步骤包含：a) 生成矢量图元；b) 图元处理。

2.2.1. 生成矢量图元

在基础纹理图像的生成过程中，矢量图元作为构建复杂纹理的基本元素，其种类丰富多样，涵盖了曲线、直线、点、规则图形以及不规则图形等。这些不同类型的矢量图元能够以各自独特的方式组合与排列，从而创造出千变万化的纹理效果。

文中主要使用参数方程生成矢量图元。在数学领域中，存在多种基础参数方程，用于描述不同几何图形的特征，常见的基础参数方程有直线、圆、椭圆和螺旋线等。其中，直线上点的位置向量通过式(1)计算。

$$r(t) = r_0 + t * d \tag{1}$$

式中，r_0 是直线上的一个已知点；d 是直线的方向向量；t 是参数，其取值范围决定直线的长度，当 t 在一定区间内变化时，根据公式(1)就会描绘出一条直线。

圆的参数方程见式(2)：

$$\begin{cases} x = h + r * \cos(t) \\ y = k + r * \sin(t) \end{cases} \tag{2}$$

式中，r 为圆的半径；(h, k)为圆心坐标，确定圆在平面直角坐标系中的位置；t 为参数，通常 $t \in [0, 2\pi]$，当 t 从 0 变化到 2 π 时，根据公式(2)可以得到一个完整的圆。

椭圆的参数方程见式(3)：

$$\begin{cases} x = a * \cos(t) \\ y = b * \sin(t) \end{cases} \tag{3}$$

式中，a 和 b 分别是椭圆的长半轴和短半轴，共同决定椭圆的形状和大小，当 $a = b$ 时，椭圆退化为圆；t 为参数，通常 $t \in [0, 2\pi]$。

抛物线的参数方程见式(4)：

$$\begin{cases} x = t \\ y = a * t^2 + b * t + c \end{cases} \tag{4}$$

式中，a，b 和 c 为常数，决定抛物线的形状、开口方向和位置，当 $a > 0$ 时，开口向上，当 $a < 0$ 时，开口向下；t 为参数，通常 $t \in [0, 2\pi]$。

螺旋线的参数方程见式(5)：

$$\begin{cases} x = r * \cos(t) \\ y = r * \sin(t) \\ z = c * t \end{cases} \tag{5}$$

式中，r 为螺旋的半径，决定螺旋线在平面上的投影大小；c 为螺旋的纵向增速，决定螺旋线在 z 轴方向上的增长速度；t 为参数，随着 t 的不断增大，螺旋线会沿着 z 轴方向逐渐上升，形成独特的螺旋形状。

为了更清晰地阐述矢量图元的生成过程，本文以半圆矢量图元为例进行详细说明。

使用公式(6)生成半圆矢量图元。

$$\begin{cases} x = R * \cos(t + \text{angle}) \\ y = R * \sin(t + \text{angle}) \end{cases} \tag{6}$$

式中，R 是圆的半径，决定半圆的大小；angle 是圆的初始化角度，决定半圆在圆周上的起始位置。

2.2.2. 图元处理

为了构建出具有特定艺术效果的矩形边框基础纹理图像，需要对 2.2.1 中获得的半圆矢量图元进行处理，如平移或旋转，进而得到基础数字纹理图像。由于本算法生成的图元均为半径相同的半圆，故只需对矢量图元进行相应角度(90˚、180˚和 270˚)的旋转，获得上下左右四边的基础图形，接着分别对基础图形进行对应方向的平移，最终获得矩形边框基础纹理图像。具体实现步骤如下：

a) 确定矩形区域，并进行区域划分

将宽度为 W，高度为 H 的图像分为以下区域，如图 2 所示。其中，*Up Pattern* 放置上边图案；*Bottom Pattern* 放置下边图案；*Left Pattern* 放置左边图案；*Right Pattern* 放置右边图案。

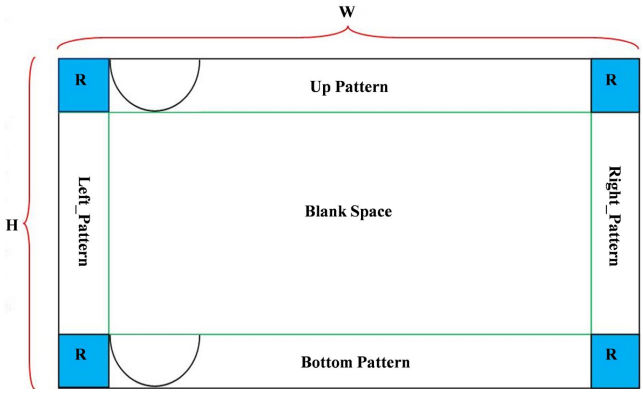


Figure 2. Schematic diagram of rectangular area
图 2. 矩形区域示意图

b) 确定坐标系

以图 2 左上角为坐标原点，向右为 x 轴正方向，向下为 y 轴正方向。

c) 确定图形关系

通过图 2 可得上面和下面的图元只在 x 方向变化，y 保持不变；左面和右面是在 y 方向变化，x 保持不变。基于 a)和 b)上面图形的 x 起始位置为 R，结束位置为 $W - R$；下面图形是由基础图元旋转 180˚得到，其旋转后的图形与上面图形一致，因此下面图形 x 起始位置和结束位置与上面图形保持一致，其 y 值进行发生变化，通过公式(7)确定图形的 y 值；同理，左面图形的 y 保持不变，x 进行变化，右面图形的 y 与左面保持一致，通过公式(8)确定 x 的值。

$$y\text{change} = y + H - R \tag{7}$$

$$x\text{change} = x + W - R \tag{8}$$

通过对半圆矢量图元的旋转、平移以及基于区域划分、坐标系确定和图形关系分析的一系列操作，最终获得矩形边框基础纹理图像。

2.2.3. 基础纹理图像生成伪代码

基于 1.2.1 生成的矢量图元和 1.2.2 中的图元处理步骤得到基础纹理图像。具体生成基础图像伪代码，如表 1 所示。

Table 1. Pseudocode for generating basic texture image
表 1. 基础纹理图像生成伪代码

input：矩形区域大小。
output：基础纹理图像。

a) *start* algorithm ()
b) *svg_width* = initialize *svg_width* ()
c) *svg_height* = initialize *svg_height* ()
d) *R* = initialize *half_circle_radius* ()
e) *gap* = initialize *gap* ()
f) *x_top*, *y_top* = *generate_half_circle* (*R*, angle = 180)
g) *x_bottom*, *y_bottom* = *generate_half_circle* (*R,* angle = 0)
h) *x_left*, *y_left* = *generate_half_circle* (*R*, angle = 90)
i) *x_right*, *y_right* = *generate_half_circle* (*R*, angle = -90*)*
j) *svg_filename* = generate_svg_filename (*R*)
k) write_points_to_svg (*svg_filename*, *svg_width*, *svg_height*, *R*, *gap*, *x_left*, *y_left*, *x_right*, *y_right*, *x_top*, *y_top*, *x_bottom*, *y_bottom*)
l) *end* algorithm ()

2.3. 随机字符化

2.3.1. 主要步骤

随机字符化的关键是进行随机选取，即在曲线进行字符化时，随机在总字符集中选择字符，每次选择字符时，均不受之前选择结果的限制，完全独立地从总字符集中进行随机抽取，因此选择的字符可以相同，也可以不同。这一过程极大地增加了字符化结果的随机性和不确定性，使得每一次生成的数字纹理图像都具有独一无二的特征。主要步骤为：

a) 设定总字符集 charters：该总字符集包括 1、2、3 等阿拉伯数字；a、b、A、B 的英文字母；α、β、γ 等希腊字母和“测”、“试”、“汉”、“字”等汉字的一种或多种。

b) 对曲线的每一段随机选取字符，直至曲线渲染完成：将曲线划分为若干段，针对每一段曲线，均独立地从 charters 中随机选择字符进行填充，直至整条曲线渲染完成。

2.3.2. 伪代码

随机字符化基础纹理图像的具体伪代码如表 2 所示。

Table 2. Pseudocode for generating basic texture image
表 2. 基础纹理图像生成伪代码

input：基础纹理图像的大小和四个基础图元起始点坐标。
output：字符化后的数字纹理图像。

a) ***start*** algorithm ()
b) *steps* = initialize half_circle_steps ()
c) *characters* = initialize characters ()
d) *line_width* = initialize *line_width* ()
e) *svg_filename* = generate_svg_filename (*R*, *steps*, *line_width*)
f) *dwg* = create_svg_drawing_object (*svg_filename*, *svg_width*, *svg_height*)
g) set_svg_viewbox (*dwg*, *svg_width*, *svg_height*)
h) *colors_set* = generate_rainbow_colors (len (*x_top*))
i) draw_characters_on_segment (*x*, *y*, *line_width*, *characters*, *colors_set*)
j) save_svg_file (*dwg*)
k) *end* algorithm ()

3. 实验

3.1. 实验目的

本实验旨在深入验证所提出的数字纹理图像生成算法的有效性，全面评估其在生成具有特殊艺术效果和防伪特性的数字纹理图像方面的性能表现。通过系统地进行实验操作和细致的结果分析，进一步明确该算法的优势，为其在实际应用中提供坚实的理论和实践依据。

3.2. 实验准备

在实验准备阶段，首先对实验所需的各项参数进行设定。

3.2.1. 矩形区域设定

对于矩形区域参数，设定宽度 W = 800 像素，高度 H = 600 像素。这一选择基于多方面考虑，从计算资源角度出发，若图像尺寸过大，在生成矢量图元、进行图元处理以及字符化操作等过程中，会消耗大量的计算资源，导致计算时间大幅延长，甚至可能超出实验设备的承载能力。而该尺寸既能保证图像具有足够的细节，满足对纹理图像艺术效果和防伪特性研究的需求，又能避免时间的过度消耗。同时，800 × 600 像素的尺寸符合常见的图像显示和处理标准，便于后续对实验结果进行展示、分析以及与其他研究成果进行比较。

3.2.2. 总字符集设定

在设定总字符集时，需充分考虑字符的多样性和复杂性，以满足不同应用场景对数字纹理图像的需求。

本算法中的总字符集 charters 包含丰富的字符类型，具体包括：“0123”等阿拉伯数字、“测试汉字”等汉字、“ABCabc”等大小写英文字母、“$\alpha\beta\gamma$”等希腊字母和它们的混合“0123 测试汉字 ABCabc$\alpha\beta\gamma$”。通过多种不同类型的字符，使生成的数字纹理图像在艺术效果上更加丰富多样，满足不同用户对个性化和创新性的需求；在防伪特性方面，增加了字符组合的可能性，提高了伪造的难度，从而增强图像的防伪能力。

3.2.3. 字符大小和字符数设定

本算法对字符大小和图元包含的字符数进行了多样化的设置。其中，字符大小 *Line_width* 分别设置为 6 像素、12 像素和 30 像素；每条图元包含的字符数 *steps* 设置为 14、24 和 39。这种多样化的设置是为了深入研究不同参数对数字纹理图像效果的影响。较小的字符大小适合表现细腻的纹理细节，在搭配较多字符数量时，能够在保持图像细腻度的同时，展现出丰富的细节，使图像在微观层面具有较高的艺术价值和防伪复杂性。而较大的字符大小则可以突出字符的个体特征，在搭配较少字符数量时，营造出简洁而强烈的视觉效果，从宏观上增强图像的辨识度和防伪独特性。

3.3. 基础纹理图像生成结果

在设定矩形区域参数 W = 800 像素，H = 600 像素后，通过本算法生成的基础纹理图像，如图 3 所示。

对图 3 进行分析，可以发现其具有一定的规律性和对称性。矩形边框的上下两边以及左右两边在形状和结构上分别呈现出对称的特点，这种对称性不仅增加了图像的美感，还体现了算法在生成过程中的严谨性。由于在图元处理过程中采用了随机平移和旋转的操作，虽然整体呈现出规则的矩形边框，但每个半圆图元的具体位置、方向和颜色仍存在一定的随机性，这使得基础纹理图像在具有规律性的同时，也具备了一定的独特性。这种规律性与随机性的结合，为后续生成具有丰富艺术效果和防伪特性的数字纹理图像奠定了坚实的基础。

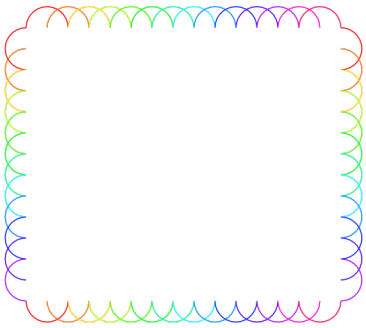


Figure 3. Base texture image
图 3. 基础纹理图像

3.4. 随机字符化纹理图像生成结果

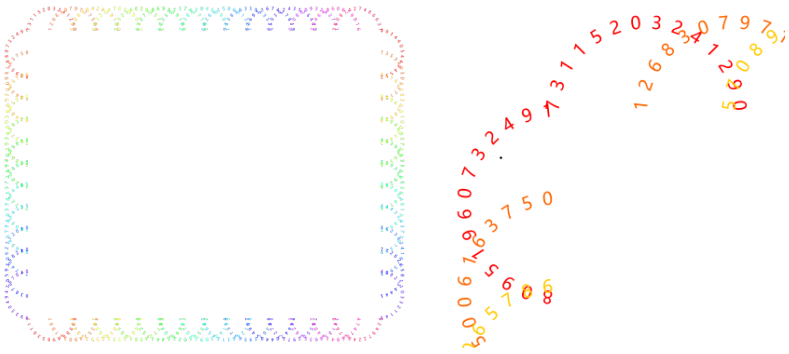
3.4.1. 不同总字符集下的结果

在实验中，通过设定不同的总字符集 *charters*，具体设定见表 3，生成了一系列具有不同字符内容的数字纹理图像。

Table 3. Set the content of the total character set
表 3. 设定总字符集内容

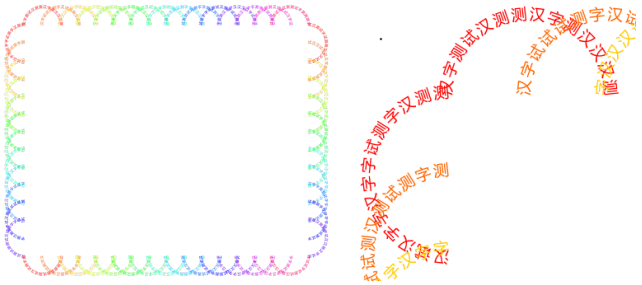
总字符集类型	内容
阿拉伯数字	“0123456789”
汉字	“测试汉字”
英文字母	“ABCDEFGHIJKLMNOPQRSTUVWXYZ；Abcdefghijklmnopqrstuvwxyz”
希腊字母	“*αβγδεζηθικλμνξοπρστυφχψω*”
混合	“0123456789” “测试汉字” “ABCDEFGHIJKLMNOPQRSTUVWXYZ；Abcdefghijklmnopqrstuvwxyz”； “*αβγδεζηθικλμνξοπρστυφχψω*”

在字符大小 *Line_width* = 30 像素和字符数 *steps* = 14 的前提下，不同 *charters* 生成的数字纹理图像如图 4~8 所示。



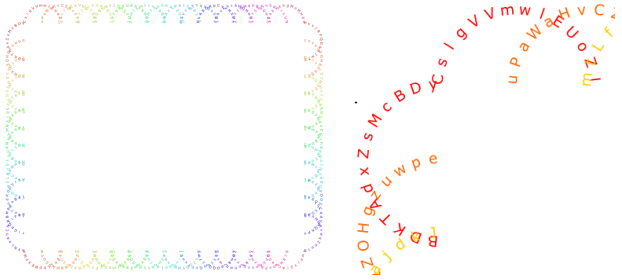
a. 整体效果图　　b. 细节图

Figure 4. Arabic numerals
图 4. 阿拉伯数字



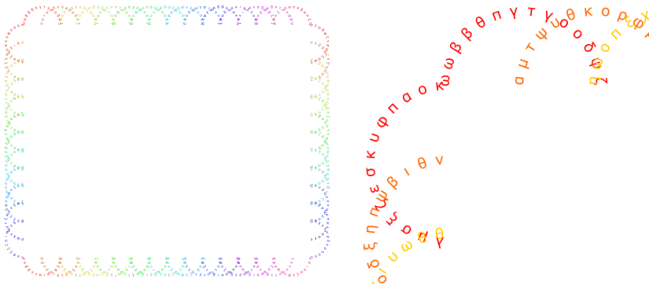
a. 整体效果图　　b. 细节图

Figure 5. Chinese characters
图 5. 汉字



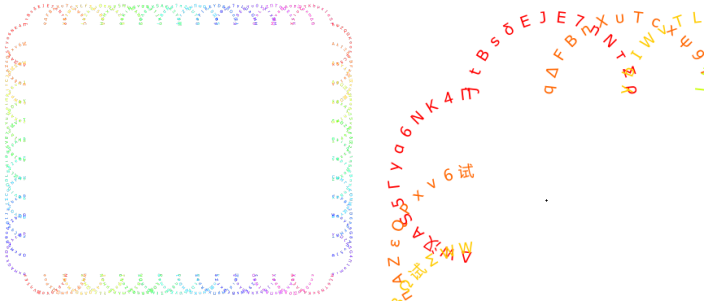
a. 整体效果图　　b. 细节图

Figure 6. English letters
图 6. 英文字母



a. 整体效果图　　b. 细节图

Figure 7. Greek alphabet
图 7. 希腊字母



a. 整体效果图　　b. 细节图

Figure 8. Mixing
图 8. 混合

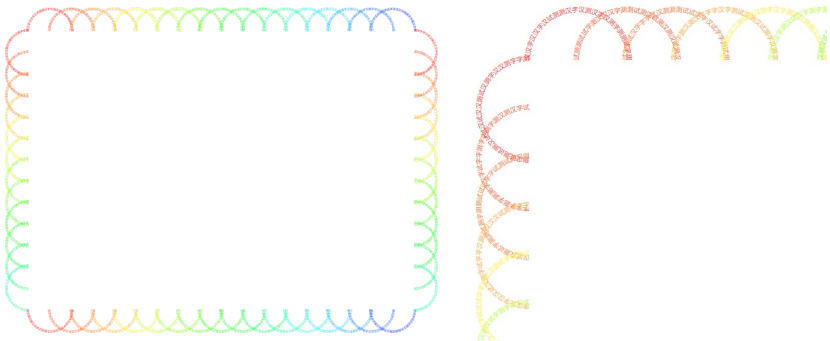
通过对图 4~8 进行分析，可以看出不同总字符集下的图像在艺术效果和防伪特性方面也存在差异。艺术效果方面，不同类型的字符具有不同的形状、笔画和风格，它们的组合方式和排列顺序决定了图像的整体艺术风格。防伪特性方面，字符集的多样性和复杂性增加了字符组合的可能性，使得伪造者难以复制出完全相同的图像。总字符集包含的字符类型越多，字符组合的可能性就越大，防伪难度也就越高。

3.4.2. 不同字符大小和数量下的结果

为了深入探究字符大小和数量参数对数字纹理图像效果的影响，在设定总字符集 *charters* = {“测试汉字”}的前提下，在实验中设置了不同的字符大小 *Line_width* (分别为 6 像素和 12 像素)和图元包含的字符数 *steps* (分别为 24 和 39)，并生成相应的数字纹理图像。不同字符大小和数量参数下生成的图像对比如图 9~12 所示。

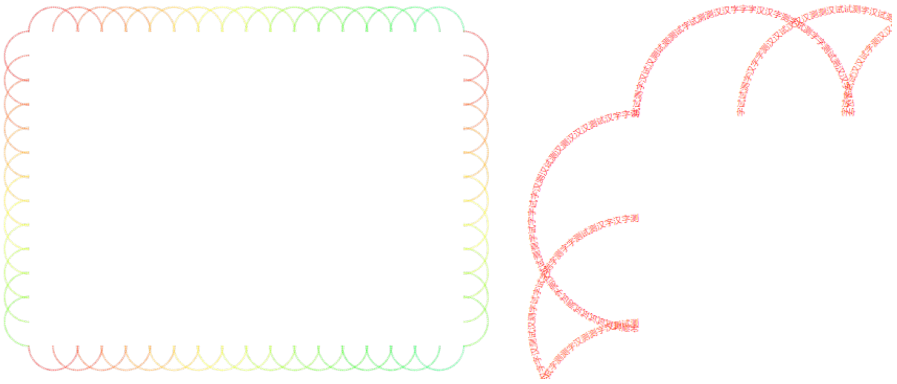
Figure 9. *Line_width* = 6
图 9. *Line_width* = 6

Figure 10. *Line_width* = 12
图 10. *Line_width* = 12



a. 整体效果图　　b. 细节图

Figure 11. *steps* = 24
图 11. *steps* = 24



(a)整体效果图　　(b)细节图

Figure 12. *steps* = 39
图 12. *steps* = 39

通过图 9~12 对比分析可以看出随着字符的增大，生成的数字纹理图像越清晰；随着图元包含字符数量的增加，图像的丰富度明显提高。如图 11 所示，当图元包含的字符数为 24 时，纹理图像相对简洁，字符之间的排列较为稀疏，整体呈现出一种简洁的视觉效果；而当字符数达到 39 时，如图 12 所示，图像的复杂程度达到较高水平，字符之间几乎紧密相连，形成了一种密集、丰富的纹理效果。此外，字符大小和数量之间存在交互作用。较小的字符大小搭配较多的字符数量，使纹理图像在保持细腻度的同时，展现出丰富的细节，呈现出独特的艺术风格；而较大的字符大小搭配较少的字符数量，则突出字符的个体特征，营造出简洁而强烈的视觉效果，在追求简洁明了风格的设计中具有应用价值。

4. 结果分析

在艺术效果上，本算法生成的数字纹理图像展现出艺术感染力，而依据纹理的唯一性和随机性这两大核心特性，在防伪领域也展现出了巨大的应用潜力。在图像生成过程中，通过参数方程生成各类矢量图元，如半圆矢量图元等，这些图元的形状、位置和方向都由参数精确控制。同时，对生成的矢量图元进行随机平移和旋转操作，使得每一个基础纹理图像都具有显著的随机性和不可预测性。即使伪造者试图模仿，也难以准确复现这些复杂的变化。

而在随机字符化阶段，从包含阿拉伯数字、英文字母、希腊字母、汉字等多种类型的丰富总字符集中随机选取字符，这一过程完全独立且不受之前选择结果的限制。每一次生成数字纹理图像时，字符的选择组合都千变万化，极大地增加了图像的不确定性。例如，假设总字符集包含 100 个不同字符，一条曲线上有 10 个字符位置，那么字符组合的可能性高达 10,010 种，如此庞大的组合数量，使得伪造者想要通过猜测或尝试来复制出完全相同的纹理图像几乎是不可能完成的任务。这种从基础纹理构建到字符化处理的多层随机性设计，为图像的防伪提供了坚实的基础保障。

5. 结论

本研究提出了一种数字纹理图像生成算法，该算法运用参数方程生成基础纹理图像，并通过随机字符化处理，最终生成具有特殊艺术效果和强大防伪特性的数字纹理图像。在算法设计方面，通过对参数方程的灵活运用，能够精确地生成各种类型的矢量图元，如半圆矢量图元等，为构建多样化的基础纹理图像奠定了坚实基础。对图元进行的随机平移和旋转操作，增加了纹理的随机性和自然感，使其更具独特性。在字符化处理过程中，从丰富的总字符集中随机选取字符，极大地增加了字符组合的可能性，赋予纹理图像独特的艺术风格和防伪功能。

实验结果清晰地表明，本算法在生成具有特殊艺术效果和防伪特性的数字纹理图像方面具有显著的有效性。在艺术效果方面，生成的图像能具有独特的创意。在防伪特性方面，图像的唯一性和随机性使得伪造变得极为困难，有效地提高了防伪性能，为产品提供更加可靠的防伪保障。

参考文献

[1] 李文秀. 自然纹理图像生成技术研究[D]: [硕士学位论文]. 哈尔滨: 哈尔滨工程大学, 2005.

[2] 李昊. 基于分形算法的轻样本图像生成研究[D]: [硕士学位论文]. 景德镇: 景德镇陶瓷大学, 2024.

[3] Mandelbrot, B.B. (1982) The Fractal Geometry of Nature, Volume 1. W. H. Freeman and Company.

[4] Bunde, A. and Havlin, S. (2012) Fractals and Disordered Systems. Springer Science & Business Media.

[5] Xu, T., Moore, I.D. and Gallant, J.C. (1993) Fractals, Fractal Dimensions and Landscapes—A Review. *Geomorphology*, **8**, 245-262. https://doi.org/10.1016/0169-555x(93)90022-t

[6] Falconer, K. (2003) Fractal Geometry: Mathematical Foundations and Applications. John Wiley & Sons. https://doi.org/10.1002/0470013850

[7] Barnsley, M.F. (2014) Fractals Everywhere. Academic Press.

[8] Hutchinson, J. (1981) Fractals and Self Similarity. *Indiana University Mathematics Journal*, **30**, 713-747. https://doi.org/10.1512/iumj.1981.30.30055

[9] Barnsley, M.F. and Harrington, A.N. (1989) The Calculus of Fractal Interpolation Functions. *Journal of Approximation Theory*, **57**, 14-34. https://doi.org/10.1016/0021-9045(89)90080-4

[10] Massopust, P.R. (1997) Fractal Functions and Their Applications. *Chaos*, *Solitons & Fractals*, **8**, 171-190. https://doi.org/10.1016/s0960-0779(96)00047-1

[11] Massopust, P.R. (2016) Fractal Functions and Wavelets. In: Massopust, P.R., Ed., *Fractal Functions*, *Fractal Surfaces*, *and Wavelets*, Elsevier, 261-327. https://doi.org/10.1016/b978-0-12-804408-7.00008-4

[12] Massopust, P.R. (1990) Fractal Surfaces. *Journal of Mathematical Analysis and Applications*, **151**, 275-290. https://doi.org/10.1016/0022-247x(90)90257-g

[13] Strichartz, R.S. (2003) Fractafolds Based on the Sierpinski Gasket and Their Spectra. *Transactions of the American Mathematical Society*, **355**, 4019-4043. https://doi.org/10.1090/s0002-9947-03-03171-4

[14] Yang, X.J., Baleanu, D. and Srivastava, H.M. (2016) Local Fractional Fourier Series. In: Yang, X.J., Baleanu, D. and Srivastava, H.M., Eds., *Local Fractional Integral Transforms and Their Applications*, Elsevier, 57-94. https://doi.org/10.1016/b978-0-12-804002-7.00002-4

[15] Barnsley, M.F., Devaney, R.L., Mandelbrot, B.B., *et al*. (1988) The Science of Fractal Images, Volume 1. Springer.

[16] Keller, J.M., Chen, S. and Crownover, R.M. (1989) Texture Description and Segmentation through Fractal Geometry. *Computer Vision*, *Graphics*, *and Image Processing*, **45**, 150-166. https://doi.org/10.1016/0734-189x(89)90130-8

[17] Barnsley, M.F. and Hurd, L.P. (1993) Fractal Image Compression. AK Peters Ltd.

[18] Pentland, A.P. (1984) Fractal-Based Description of Natural Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, 661-674. https://doi.org/10.1109/tpami.1984.4767591

[19] Draves, S. (2005) The Electric Sheep Screen-Saver: A Case Study in Aesthetic Evolution. In: *Lecture Notes in Computer Science*, Springer, 458-467. https://doi.org/10.1007/978-3-540-32003-6_46

[20] Ebert, D.S., Kenton Musgrave, F., Peachey, D., Perlin, K. and Worley, S. (2003) Preface. In: Ebert, D.S., Peachey, D., Worley, S., Hart, J.C. Peachey, D., Worley, S., *et al*., Eds., *Texturing and Modeling: A Procedural Approach*, Elsevier, 20-23. https://doi.org/10.1016/b978-155860848-1/50029-2

[21] Prusinkiewicz, P. and Lindenmayer, A. (2012) The Algorithmic Beauty of Plants. Springer Science & Business Media.

[22] Campbell, R.D. (1996) Describing the Shapes of Fern Leaves: A Fractal Geometrical Approach. *Acta Biotheoretica*, **44**, 119-142. https://doi.org/10.1007/bf00048419

[23] Hartvigsen, G. (2000) The Analysis of Leaf Shape Using Fractal Geometry. *The American Biology Teacher*, **62**, 664-669. https://doi.org/10.2307/4451007

[24] Godin, C. and Ferraro, P. (2010) Quantifying the Degree of Self-Nestedness of Trees: Application to the Structural Analysis of Plants. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, **7**, 688-703. https://doi.org/10.1109/tcbb.2009.29

[25] Perlin, K. (1985) An Image Synthesizer. *Proceedings of the* 12*th Annual Conference on Computer Graphics and Interactive Techniques*, New York, 1 July 1985, 287-296. https://doi.org/10.1145/325334.325247

[26] 曹鹏, 赵珊珊. 伪随机矢量数字纹理图像、生成与检测方法[P]. 中国专利, 202410828667.4. 2024-11-12.